A feasible theory of truth over combinatory algebra

Sebastian Eberhard[1]

Institut für Informatik und angewandte Mathematik, Universität Bern, Neubrückstrasse 10, CH-3012 Bern, Switzerland.

Abstract

We define an applicative theory of truth T_{PT} which proves totality exactly for the polynomial time computable functions. T_{PT} has natural and simple axioms since nearly all its truth axioms are standard for truth theories over an applicative framework. The only exception is the axiom dealing with the word predicate. The truth predicate can only reflect elementhood in the words for terms that have smaller length than a given word. This makes it possible to achieve the very low proof-theoretic strength. Truth induction can be allowed without any constraints. For these reasons the system T_{PT} has the high expressive power one expects from truth theories. It allows embeddings of feasible systems of explicit mathematics and bounded arithmetic.

The proof that the theory T_{PT} is feasible is not easy. It is not possible to apply a standard realisation approach. For this reason we develop a new realisation approach whose realisation functions work on directed acyclic graphs. In this way, we can express and manipulate realisation information more efficiently.

Keywords: Polytime computability, applicative theories, truth theories
2010 MSC: 03F03, 03F50

1. Introduction

The theory of truth T_{PT} introduced in this paper is based on an applicative ground language for operations in the sense of combinatory logic; operations can freely be applied to other operations and strong principles of recursion are available due to the known expressive power of combinatory algebras. The first order applicative base describes the operational core of Feferman's explicit mathematics, cf. [13, 14, 15]. The notion of a partial, self-referential predicate of truth is rooted in Frege's seminal work. Theories which expand an applicative core with such a truth predicate are

Email address: eberhard@iam.unibe.ch (Sebastian Eberhard)
[1]Research supported by the Swiss National Science Foundation.

introduced in the work of Aczel [1] and Beeson [2]. Similar theories to the one introduced in this paper were developed in Cantini [4, 5] and Kahle [24, 25]. For important results in the realm of truth theories over *arithmetical* ground theories, see e.g. Feferman's [16, 18], Friedman and Sheard [21]. For a comprehensive overview and newer results see Halbach [22].

The theory T_{PT} that will be developed in this paper can be seen as feasible analogue of Cantini's theory of truth in [6]. As Cantini's theory, also T_{PT} contains unrestricted truth induction and natural axioms for compositional truth. The only difference between the two theories is that T_{PT} reflects only elementhood in the words for terms that have smaller length than a given word. This restriction is responsible for its very low proof theoretic strength. The idea to restrict the reflection of elementhood in the words in order to obtain weak theories was also used in explicit mathematics where types for the initial segments of the words were introduced by Spescha and Strahm in [28, 29]. The system PETJ, introduced there, can indeed be seen as analogue of the theory T_{PT} in explicit mathematics. PETJ was proven to be feasible by Probst in [26] using non-standard models. The close connection of T_{PT} and PETJ is established by Strahm and the author in [12] where mutual embeddings of T_{PT} and a - from the point of expressive power - strengthened version of PETJ are demonstrated. Presupposing the feasibility of T_{PT} this yields a new proof of the feasibility of PETJ. Indeed, embeddings into T_{PT} are possible for many other feasible systems such as Buss' S^1_2 [3] or Cook and Urquart's PV^ω [9]. That T_{PT} proves totality for all polynomial time computable functions follows from these embeddings, or also directly using their well-known function algebra description developed by Cobham in [8].

In this paper, we will focus on the proof of the upper bound of T_{PT}. Upper bounds for weak applicative theories are usually established using realisation techniques as developed by Cantini in [6] and Strahm in [31]. This is because for most of the analysed theories, embeddings into bounded arithmetic do not seem to be possible because equality of lambda terms is already Σ_1 complete. The upper bound computation of T_{PT} is difficult because the usual realisation approach does not work. A new realisation approach will be developed which uses directed acyclic graphs to store and manipulate realisation information more efficiently. This approach also allows to find upper bounds for the corresponding theories of explicit mathematics, but can be motivated best for T_{PT}.

We conclude the introduction with a detailed outline of the paper. In Section 2, we will introduce the basic applicative framework of T_{PT}, which was developed by Strahm in [31]. Strahm's system uses a predicate W for binary words instead

of a predicate N for natural numbers as ground type, which allows to state weak induction principles in a very natural way. As usual for theories of truth, we always work in a total setting. In Section 3, we introduce the theory T_{PT} which extends the applicative axioms with a compositional truth predicate T and the principle of truth induction. We will discuss some of the theorems of T_{PT}.

The rest of the paper is devoted to the upper bound proof of T_{PT}. First, we introduce the realisation approach developed by Cantini in [6] which allowed him to find upper bounds for theories of truth with additional principles such as choice and uniformity. We do so because the new realisation approach is based on Cantini's approach. It will also be shown were Cantini's approach fails when it is applied to T_{PT}, which also motivates the new approach. In Section 5, we give its technical details. We define a special set of directed acyclic graphs with multiple edges, and explain how they carry realisation information.

In section 6, we show how this approach can be used to find the upper bound for an intuitionistic version $\mathsf{T}^i_{\mathsf{PT}}$ of T_{PT}. The restriction to intuitionistic logic allows us to present the ideas more transparently. Nevertheless, the approach could easily be adapted, in a similar way as presented in Strahm's [31], to deal with classical logic. Most of the work has to be done to realise the induction rule which is realised, as usual, using bounded recursion. An important difference to realisations of other applicative theories of polynomial strength, such as PT introduced by Strahm in [31] or PETJ, is that this bound cannot be constructed directly from the form of the induction formula and the realisation function for a special induction premise. Instead the bound must be established using bounding conditions which can be proved to hold for all used realisation functions by induction on the depth of the corresponding proof. We conclude this lengthy section by sketching how the approach could be adapted to realise the classical version of T_{PT}. Finally, in section 7, we mention our current research and related research.

2. The basic applicative framework

The theory T_{PT} that is studied in this paper is based on an applicative base theory which includes the axioms for a total[2] combinatory algebra and a basic data type

[2]We work in a total setting because this is the usual framework for theories of truth. Otherwise, problems with the reflection of negated atoms containing undefined terms occur. Since our truth predicate does not reflect negative formulas anyway, partiality is not problematic in our case. The partial version of T_{PT} has equal strength as T_{PT} since it allows the same direct lower bound proof. A detailed discussion of *partial* applicative theories of truth can be found in Kahle [23].

W which is interpreted as the set $\mathbb{W} = \{0,1\}^*$ of binary words in the standard interpretation. As usual, $\subseteq$ denotes the relation of being an initial subword, and $\leq$ the relation of having a smaller length. By the length $|w|$ of a word w, we denote the number of zeros and ones it is build of.

2.1. The applicative language L

Our basic language L is a first order language for the logic of partial terms which includes:

- variables $a, b, c, x, y, z, u, v, f, g, h, \ldots$
- constants k, s, p, p_0, p_1, d_W, ϵ, s_0, s_1, p_W, $\mathsf{c}_\subseteq$, $*$, $\times$
- relation symbols $=$ (equality), W (binary words)
- the binary function constant $\circ$ (application)

The meaning of the constants will become clear in the next paragraph.

The terms $(r, s, t, p, q, \ldots)$ are inductively generated from the variables and constants by means of application. So if s and t are terms then also $\circ(s,t)$. The formulas $(A, B, C, \ldots)$ of L are given as the closure of the atoms $s = t$, $\mathsf{W}(s)$ under negation the connectors $\wedge, \vee$ and the quantifiers $\exists, \forall$. We assume the following standard abbreviations and syntactical conventions:

$$\begin{aligned}
t_1 t_2 \ldots t_n &:= (\ldots(t_1 \circ t_2) \circ \cdots \circ t_n) \\
s(t_1, \ldots, t_n) &:= s t_1 \ldots t_n \\
t \in \mathsf{W} &:= \mathsf{W}(t) \\
t : \mathsf{W}^k \to \mathsf{W} &:= (\forall x_1 \ldots x_k \in \mathsf{W}) t x_1 \ldots x_k \in \mathsf{W} \\
\mathsf{c}_\leq(s,t) &:= \mathsf{c}_\subseteq(1\times s, 1\times t) = 0 \\
s \leq_\mathsf{W} t &:= \mathsf{c}_\leq(s,t) \wedge s \in \mathsf{W}
\end{aligned}$$

In the following we often write $A[\vec{x}]$ in order to indicate that the variables $\vec{x} = x_1, \ldots, x_n$ may occur free in A. Finally, let us write $\overline{w}$ for the canonical closed L term denoting the binary word $w \in \mathbb{W}$.

2.2. The basic theory of operations and words B

The applicative base theory B has been introduced in Strahm [30, 31]. We present a total version of this theory and can therefore use classical logic. The non-logical axioms of B include:

- partial combinatory algebra:

$$\mathsf{k}xy = x, \qquad \mathsf{s}xyz = xz(yz)$$

- pairing p with projections p_0 and p_1
- defining axioms for the binary words W with ϵ, the binary successors s_0, s_1 and the predecessor p_W
- definition by cases d_W on W
- initial subword relation $\mathsf{c}_\subseteq$
- word concatenation $*$, word multiplication $\times$[3]

These axioms are fully spelled out in Strahm's [30, 31].

Let us remind the reader of the standard open term model $\mathcal{TM}$ of B: Take the universe of open λ terms and consider the usual reduction of the extensional untyped lambda calculus $\lambda\eta$, augmented by suitable reduction rules for the constants other than k and s. Interpret application as juxtaposition. Two terms are equal if they have a common reduct and W denotes those terms that reduce to a "standard" word $\overline{w}$.

3. The system T_PT

The system T_PT contains a predicate T that mimics the properties of positive truth, e.g. the properties of truth restricted to formulas not containing negation. The axiomatisation of this predicate relies on a coding mechanism for formulas. In the applicative framework, we code formulas using new constants designating logical operations.

3.1. The language L_T *of positive truth*

The (first order) language of T_PT is an extension of the language L by

- a new unary predicate symbol T for *truth*
- new individual constants $\doteq$, $\dot{\mathsf{W}}$, $\dot{\wedge}$, $\dot{\vee}$, $\dot{\exists}$, $\dot{\forall}$

The new constants allow the coding of negation-free formulas, called positive formulas in the following. We will use infix notation for $\doteq, \dot{\wedge}$ and $\dot{\vee}$.

[3] $x \times y$ signifies the length of y fold concatenation of x with itself; note that we use infix notation for $*$ and $\times$.

3.2. The axioms and rules of T_{PT}

The theory T_{PT} with language L_{T} is an extension of the *total* version of B by compositional truth axioms and truth induction. Accordingly, its underlying logic is simply first order classical predicate logic.

Compositional truth

(C1) $\mathsf{T}(a \doteq b) \leftrightarrow a = b$

(C2) $a \in \mathsf{W} \rightarrow (\mathsf{T}(\dot{\mathsf{W}}ab) \leftrightarrow b \leq_{\mathsf{W}} a)$

(C3) $\mathsf{T}(a\dot{\vee}b) \leftrightarrow \mathsf{T}(a) \vee \mathsf{T}(b)$

(C4) $\mathsf{T}(a\dot{\wedge}b) \leftrightarrow \mathsf{T}(a) \wedge \mathsf{T}(b)$

(C5) $\mathsf{T}(\dot{\exists}a) \leftrightarrow \exists x \mathsf{T}(ax)$

(C6) $\mathsf{T}(\dot{\forall}a) \leftrightarrow \forall x \mathsf{T}(ax)$

Additionally, we have unrestricted truth induction.

Truth Induction

$$\mathsf{T}(a\epsilon) \wedge (\forall x \in \mathsf{W})(\mathsf{T}(ax) \rightarrow \mathsf{T}(a(\mathsf{s}_0 x)) \wedge \mathsf{T}(a(\mathsf{s}_1 x))) \rightarrow (\forall x \in \mathsf{W})(\mathsf{T}(ax))$$

3.3. Theorems of T_{PT}

Let us give the set of formulas for which the Tarski biconditionals hold.

Definition 1 *Let A be a positive L_{T} formula and u be a variable not occurring in A. Then the formula A^u is obtained by replacing each subformula of the form $t \in \mathsf{W}$ of A by $t \leq_{\mathsf{W}} u$.*

The following lemma can be proved by an easy external induction on the complexity of A.

Lemma 2 *Let A be a positive L_{T} formula. Then, we have*

$$\mathsf{T}_{\mathsf{PT}} \vdash u \in \mathsf{W} \rightarrow (\mathsf{T}(\langle A^u \rangle) \leftrightarrow A^u),$$

where $\langle A^u \rangle$ denotes the obvious code of A^u, see [12] for details.

The strength of theories weaker than Peano arithmetic is usually measured by giving their provably total functions. We use the standard definition of provable totality in the applicative setting.

Definition 3 *A function* $F : \mathbb{W}^n \to \mathbb{W}$ *is called provably total in an* L_{T} *theory* T*, if there exists a closed* L *term* t_F *such that*

(i) $\mathsf{T} \vdash t_F : \mathsf{W}^n \to \mathsf{W}$ *and, in addition,*

(ii) $\mathsf{T} \vdash t_F \overline{w}_1 \cdots \overline{w}_n = \overline{F(w_1, \ldots, w_n)}$ *for all* $w_1, \ldots, w_n$ *in* $\mathbb{W}$*.*

We can easily show that all polynomial time computable functions are provably total in T_{PT}. This is done by an external induction on the rank of the function relative to Cobham's function algebra description given in Clote [7].

The theory PETJ of explicit mathematics of polynomial strength which is defined and analysed by Spescha and Strahm in [27, 28, 29] can be embedded into T_{PT} by a standard embedding which is illustrated by Strahm and the author in [12]. This gives an alternative proof of the lower bound. The embedding is straightforward and uses the well-known correspondence between sets and unary predicates: Set constants have to be defined by terms which formulate their elementhood conditions, or by terms that do so when applied to a suitable number of arguments, respectively. The extension of PETJ by the axiom that everything is a name can be embedded using the same approach.

3.4. Sequent style formulation of $\mathsf{T}^i_{\mathsf{PT}}$

As mentioned before, we will detail the upper bound proof for the intuitionistic version $\mathsf{T}^i_{\mathsf{PT}}$ of T_{PT}. The realisation approach is best formulated for systems in sequent style, and it is routine to formulate $\mathsf{T}^i_{\mathsf{PT}}$ or T_{PT} in this way. We can assume that the axioms contain only positive formulas. Induction is formulated as a rule with positive main formulas in the usual way. Because of this restrictive formulation of the sequent calculus, a standard cut elimination argument yields the following lemma.

Lemma 4 *Let* T *be the theory* T_{PT} *or* $\mathsf{T}^i_{\mathsf{PT}}$*. Let* Γ, D *be a sequence of positive formulas such that* $\mathsf{T} \vdash \Gamma \Rightarrow D$*. Then there exists a* T *proof of* $\Gamma \Rightarrow D$ *that contains only positive formulas.*

4. The standard realisation approach

We denote by standard realisation approach the realisation technique executed in Cantini [6] for weak theories of truth and in Strahm [31] for feasible applicative theories.

4.1. Cantini's realisation relation

Our version of Cantini's realisation relation allows to discriminate realisers of different atoms, disjunctions and conjunctions. All relevant properties are unchanged by these modifications.

We will define the realisation relation with the help of an abstract derivability relation $d \vdash^m t$ where $d \in \mathbb{W}$, $m \in \omega$, and t is an arbitrary term, by means of a set of introduction rules, where m measures the length of proof. Assume that $\ulcorner = \urcorner, \ulcorner \mathsf{T} \urcorner, \ulcorner \mathsf{W} \urcorner, \ulcorner \wedge \urcorner, \ulcorner \vee \urcorner$ are different words. We denote in the following the equality of the terms s, t in the standard open term model by $s = t$. We also assume that $\langle \cdots \rangle$ denotes a polynomial time computable tupling function for arbitrary arity with the property that tuples of different arities are different. An example for such a pairing function is given in Clote's [7].

- $\dot{=}$-rule
$$\frac{t = a \dot{=} b \quad a = b}{\langle \ulcorner \mathsf{T} \urcorner, \epsilon \rangle \vdash^m t \text{ for } m \in \mathbb{N}}$$

- $\dot{\mathsf{W}}$-rule
$$\frac{t = \dot{\mathsf{W}} r s \quad s = \overline{\rho} \quad \mathsf{c}_{\leq}(s, r)}{\langle \ulcorner \mathsf{T} \urcorner, \rho \rangle \vdash^m t \text{ for } m \in \mathbb{N}}$$

- $\dot{\vee}$-rule
$$\frac{t = r \dot{\vee} s \quad d \vdash^n r \quad (\text{or } d \vdash^n s)}{\langle \ulcorner \vee \urcorner, 0, d \rangle \vdash^m t \quad (\text{or } \langle \ulcorner \vee \urcorner, 1, d \rangle \vdash^m t) \text{ for } n < m}$$

- $\dot{\wedge}$-rule
$$\frac{t = r \dot{\wedge} s \quad d \vdash^{n_1} r \quad e \vdash^{n_2} s}{\langle \ulcorner \wedge \urcorner, d, e \rangle \vdash^m t \text{ for } n_1, n_2 < m}$$

- $\dot{\forall}$-rule (assume $x \notin FV(rt)$)
$$\frac{t = \dot{\forall} r \quad d \vdash^n rx}{d \vdash^m t \text{ for } n < m}$$

- $\dot{\exists}$-rule
$$\frac{t = \dot{\exists} r \quad d \vdash^n rq \text{ for some } q}{d \vdash^m t \text{ for } n < m}$$

We abbreviate $(\exists m)(d \vdash^m t)$ as $d \vdash t$. Now we are in the position to define the realisation relation for all positive formulas of L_{T}. We denote $\beta\eta$ equality between

terms s, t below by $s = t$.

$$
\begin{array}{lll}
\rho \;\mathfrak{R}\; \mathsf{T}(t) & \text{iff} & \rho \vdash t \\
\rho \;\mathfrak{R}\; \mathsf{W}(t) & \text{iff} & \rho = \langle \rho_0, \rho_1 \rangle \wedge t = \overline{\rho_1} \wedge \rho_0 = \ulcorner \mathsf{W} \urcorner \\
\rho \;\mathfrak{R}\; (t_1 = t_2) & \text{iff} & \rho = \langle \rho_0, \rho_1 \rangle \wedge \rho_1 = \epsilon \wedge t_1 = t_2 \wedge \rho_0 = \ulcorner \mathsf{W} \urcorner \\
\rho \;\mathfrak{R}\; (A \wedge B) & \text{iff} & \rho = \langle \rho_0, \rho_1, \rho_2 \rangle \wedge \rho_0 = \ulcorner \wedge \urcorner \wedge \rho_1 \;\mathfrak{R}\; A \wedge \rho_2 \;\mathfrak{R}\; B, \\
\rho \;\mathfrak{R}\; (A \vee B) & \text{iff} & \rho = \langle \rho_0, \rho_1, \rho_2 \rangle \wedge \rho_0 = \ulcorner \vee \urcorner \wedge (\rho_1 = 0 \wedge \rho_2 \;\mathfrak{R}\; A) \vee \\
 & & \qquad\qquad (\rho_1 = 1 \wedge \rho_2 \;\mathfrak{R}\; B), \\
\rho \;\mathfrak{R}\; (\forall x) A(x) & \text{iff} & \rho \;\mathfrak{R}\; A(u) \text{ for a fresh variable } u, \\
\rho \;\mathfrak{R}\; (\exists x) A(x) & \text{iff} & \rho \;\mathfrak{R}\; A(t) \text{ for some term } t.
\end{array}
$$

This definition assures that we can discriminate realisers of atoms of the form $\mathsf{W}(t)$ and $\mathsf{T}(\dot{\mathsf{W}}st)$, which is crucial for the new realisation approach.

4.2. Not treatable sequent by standard realisation approach

In the following, we derive a sequent in T_{PT} for which there is no polynomial time computable realisation function relative to the standard approach. In T_{PT} we have totality and the λ-theorem holds because it includes B. Therefore, there is a closed term r which satisfies the following recursion equations for any $w \in \mathsf{W}$.

- $r(\epsilon) = 0 \dot{=} 0$
- $r(\mathsf{s}_i w) = r(w) \;\dot{\wedge}\; r(w)$

Using logical and applicative axioms, C1, C4 and truth induction we get:

$$\mathsf{T}_{\mathsf{PT}} \vdash x \in \mathsf{W} \Rightarrow \mathsf{T}(rx)$$

But we can not find a (standard) polynomial time computable realisation function for this sequent: Internal as well as external conjunctions are realised (roughly) by a pair which contains the realisers of both conjuncts. Therefore, using natural assumptions about the pairing function, realisation functions of the above displayed sequent must grow exponentially.

4.3. Inefficiencies in the standard realisation approach

Two inefficiencies of the standard realisation approach, which are closely related, will be demonstrated in the following. We will overcome them using the new realisation approach.

Let us look first at the realisers of the formulas $\mathsf{T}(r\overline{w})$ for the function r defined as before and $w \in \mathbb{W}$. Intuitively, these realisers do not contain much information, they just contain, repeatedly paired, the information ϵ. The realisers only grow that fast in w because we ask for realisation information for each internal conjunct of each internal conjunction of $r\overline{w}$ even if two such conjuncts always have the same realiser. Our formalism will take advantage of this by allowing that the same piece of realisation information can be used for several (internal) subformulas.

Another closely related source of inefficiency in the standard realisation approach can be demonstrated for the realisation of the conclusion of the cut rule. Let the used cut rule have the following form.

$$\frac{\Gamma \Rightarrow A \qquad \Gamma, A \Rightarrow D}{\Gamma \Rightarrow D}$$

We assume realisation functions p and q for the premises. To produce a realiser of D, we will first produce realisation information for A, and add this information to the tuple of realisers of Γ. Then we will apply the realisation function q. This is inefficient because realisation information that is necessary for A may already be contained in the realisers of Γ. This means that we apply the realisation function q to an input that is larger than it has to be. The formalism developed in this section allows to use the same realisation information for the subformulas of *several* formulas in a sequence and therefore overcomes this inefficiency.

5. The new formalism

5.1. Sketch of the new approach

In the previous session, we have seen examples of inefficiency since the same realisation information was produced several times instead of shared. A natural way to allow the reuse of information is the use of directed acyclic graphs with multiple edges (dagme). In the new approach, we interpret the vertices of such graphs as addresses under which realisation information is stored. E.g. the dagme with vertices v_0, v_1 containing two edges from v_1 to v_0 stores at vertex v_1 a pair whose components both are stored at vertex v_0.

$$v_n \rightrightarrows v_{n-1} \rightrightarrows \cdots \rightrightarrows v_0(\epsilon)$$

Following this informal interpretation the dagme above, whose vertex v_0 is indexed by ϵ, is interpreted as follows: Because v_n has two outgoing edges, it stores a pair. Since both edges lead to v_{n-1}, both components contain the content stored at vertex v_{n-1}. The content stored at v_{n-1} is calculated analogously. This interpretation

finally yields that the standard realiser of $\mathsf{T}(r\overline{w})$ for $w \in \mathbb{W}$ with $|w| = n$ is stored at v_n since at vertex v_0, the empty word ϵ is stored[4].

Note that the dagme given above has linear size in n, in contrast to the standard realiser of $\mathsf{T}(r\overline{w})$ with $|w| = n$, because it allows to use the same pieces of information for several internal subformulas. Note also that for all words $v \subseteq w$ the realiser of $\mathsf{T}(r\overline{v})$ is simultaneously stored at vertex m where $|v| = m$.

5.2. Realisation dags

Let us now define precisely the special sort of dagmes relevant for the realisation approach.

Definition 5 (Realisation dags) *A realisation dag (RD) α is a finite dagme with vertices V and edges E fulfilling the following conditions.*

- *Each $v \in V$ has at most two outgoing edges.*
- *Each $v \in V$ is indexed by a unique word (its address). We call the set of these indices the addresses of α.*
- *For each $v \in V$, if v has two outgoing edges, one of these edges is indexed by* 0 *and the other one by* 1.
- *For each $v \in V$, if v has exactly one outgoing edge, this edge is indexed by* 0, 1 *or it is not indexed.*
- *For each $v \in V$, if v is a leaf, it is indexed (in addition to its address) by $\langle \ulcorner \mathsf{W} \urcorner, w \rangle$ or $\langle \ulcorner \mathsf{T} \urcorner, w \rangle$ for some word w.*
- *α does not contain other indices.*

The rational behind this definition becomes clear by looking at the definition of the function con which allows to construct standard realisers from realisation dags. Depending on the number and the indices of the outgoing edges of an input vertex v of an input realisation dag α, con constructs a realiser for a conjunction, disjunction, or for an atom.

Notations 6 *For an address c of a realisation dag α, we write $v_\alpha(c)$ for its vertex indexed by c. We drop the subindex α if the realisation dag is clear from the context.*

[4] In this informal description, we ignore the fact that conjunctions are realised by triples with first component $\ulcorner \wedge \urcorner$.

Definition 7 (Construction function con) *Let α be a realisation dag and c a word. Then the function $\mathsf{con}(\alpha, c)$ is defined recursively as follows.*

If c is not an address of α, we return a fixed word ε (error) which is not a realiser of any formula. In all other cases, we execute the following definition by cases.

Case 1 There are two outgoing edges from $v(c)$ in α. The edge indexed by i leads to $v(d_i)$ for $0 \leq i \leq 1$:

$$\mathsf{con}(\alpha, c) := \langle \ulcorner \wedge \urcorner, \mathsf{con}(\alpha, d_0), \mathsf{con}(\alpha, d_1) \rangle.$$

Case 2 There is a unique outgoing edge from $v(c)$ in α. This edge is indexed by $0 \leq i \leq 1$ and leads to $v(d)$:

$$\mathsf{con}(\alpha, c) := \langle \ulcorner \vee \urcorner, i, \mathsf{con}(\alpha, d) \rangle.$$

Case 3 There is a unique outgoing edge from $v(c)$ in α. This edge is not indexed and leads to $v(d)$:

$$\mathsf{con}(\alpha, c) := \mathsf{con}(\alpha, d).$$

Case 4 $v(c)$ is a leaf indexed by $\langle \ulcorner \mathsf{W} \urcorner, w \rangle$:

$$\mathsf{con}(\alpha, c) := \langle \ulcorner \mathsf{W} \urcorner, w \rangle.$$

Case 5 $v(c)$ is a leaf indexed by $\langle \ulcorner \mathsf{T} \urcorner, w \rangle$:

$$\mathsf{con}(\alpha, c) := \langle \ulcorner \mathsf{T} \urcorner, w \rangle.$$

Notations 8 *In the following, we often abbreviate the word $\overbrace{0 \cdots 0}^{n \text{ times}}$ by n. We abbreviate indices of the form $\langle \ulcorner \mathsf{W} \urcorner, w \rangle$ for $w \in \mathbb{W}$ by w. We use Greek letters to refer to realisation dags.*

Example 9 *Let α be the following realisation dag which extends the dagme presented on page 10 by indices. It contains $n + 1 \in \mathbb{N}$ vertices, indexed by the words $0, 1, 2, \cdots, n$. For each $0 < i \leq n$, $v(i)$ has two outgoing edges, both lead to $v(i-1)$. Finally, the leaf $v(0)$ is indexed by ϵ.*

Let us calculate $\mathsf{con}(\alpha, n)$. (We suppress pairing with $\ulcorner \wedge \urcorner$.)

$$\mathsf{con}(\alpha, n) = \langle \mathsf{con}(\alpha, n-1), \mathsf{con}(\alpha, n-1) \rangle =$$
$$\big\langle \langle \mathsf{con}(\alpha, n-2), \mathsf{con}(\alpha, n-2) \rangle, \langle \mathsf{con}(\alpha, n-2), \mathsf{con}(\alpha, n-2) \rangle \big\rangle = \cdots$$

This calculation finally delivers the standard realiser of $\mathsf{T}(r\overline{n})$.

Let us now define formally, how RDs are used to realise sequences of formulas.

Definition 10 (Realisation relation) *Let* $A_1, \ldots, A_n$ *be a sequence of positive formulas. Let* α *be a RD. Let* $b : \mathbb{W} \times \mathbb{W} \to \mathbb{W}$ *be a polynomial time computable function. Then the following holds.*

$$\alpha \ \mathfrak{r}_b \ A_1, \ldots, A_n \ :\Leftrightarrow \ \textit{For all } i \textit{ with } 1 \leq i \leq n : \mathsf{con}(\alpha, b(\alpha, i)) \ \mathfrak{R} \ A_i$$

From now on, in such a context, b is called an address finder. We call the words denoted by i with $1 \leq i \leq n$ *its relevant inputs.*

Note that the role of b is to find the addresses at which standard realisers for the formulas $A_1, \cdots, A_n$ are stored. It is easy to prove the usual elementary properties for the above defined realisation relation since it is based on $\mathfrak{R}$ which has the same properties.

Lemma 11 *Let b be an address finder. Let* $A_1, \ldots, A_n$ *be a sequence of positive formulas. We let* $\vec{s} = \vec{t}$ *abbreviate* $s_0 = t_0 \wedge \cdots \wedge s_m = t_m$. *Then the following assertions hold.*

- $\alpha \ \mathfrak{r}_b \ A_1, \ldots, A_n[\vec{x}]$ *implies* $\alpha \ \mathfrak{r}_b \ A_1, \ldots, A_n[\vec{s}]$ *for all* $\vec{s}$.
- $\alpha \ \mathfrak{r}_b \ A_1, \ldots, A_n[\vec{s}]$ *and* $\mathcal{TM} \vDash \vec{s} = \vec{t}$ *implies* $\alpha \ \mathfrak{r}_b \ A_1, \ldots, A_n[\vec{t}]$ *for all* $\vec{s}, \vec{t}$.

To realise the theory $\mathsf{T}^i_{\mathsf{PT}}$ using our realisation approach, we define a realisation function $f_{\Gamma \Rightarrow D}$ for each sequent $\Gamma \Rightarrow D$ provable in $\mathsf{T}^i_{\mathsf{PT}}$. $f_{\Gamma \Rightarrow D}$ will take as input a RD realising Γ and yield as output a RD realising the sequence Γ, D. T_{PT} is realised analogously. Of course, the computational complexity of the realisation functions will be crucial. To apply notations of complexity theory to them, we implicitly interpret RDs as words, assuming a coding. In the following, we sketch how this coding works and introduce at the same time an efficient notation system for RDs.

Notations 12 *Let* α *be a RD with vertices* V *and edges* E. *Then* α *is denoted by the finite set* S *build as follows.*

- *For each vertex* $v(c) \in V$, *if* $v(c)$ *has an edge indexed by* $0 \leq i \leq 1$ *leading to* $v(d) \in V$, S *contains the string* $c \xrightarrow{i} d$.
- *For each vertex* $v(c) \in V$, *if* $v(c)$ *has a non-indexed edge leading to* $v(d)$, S *contains the string* $c \to d$.

- *For each leaf $v(c) \in V$ indexed by i, S contains the string $c : i$.*

The elements of S are denoted as RD parts of α. In the following, we identify realisation dags α and their representation as finite set S. For S containing exactly RD parts $s_1, \cdots, s_n$, we write $s_1/\cdots/s_n$ in the following.

Using a natural coding, the elements of S as well as S itself are considered as words. In the following, when we talk about functions on RDs, we implicitly assume that these functions are defined on words using the above mentioned coding function. They are assumed to output ε if one of their arguments intended to code a RD does not do so. It can be checked easily, whether some word codes a *RD*, therefore this assumption is not problematical even when working in a polynomial time setting.

5.3. Important functions on RDs

We define functions on RDs which are crucial for the upper bound proof of $\mathsf{T}^i_{\mathsf{PT}}$ presented in the next section.

First, we define a function $\mathsf{con}_{\mathsf{W}}$ which allows to extract realisation information of the form $\langle \ulcorner \mathsf{W} \urcorner, w \rangle$ for $w \in \mathbb{W}$ from its input α. The following definition is needed for this purpose.

Definition 13 (Reachable address relation) *Let c, d be addresses of a RD α. The address d is reachable from c relative to α, i.e. $R_\alpha(c, d)$ holds, exactly if there is a path from $v(c)$ to $v(d)$ in α.*

Definition 14 ($\mathsf{con}_{\mathsf{W}}$) *The function $\mathsf{con}_{\mathsf{W}} : \mathbb{W} \times \mathbb{W} \to \mathbb{W}$ is defined by the following algorithm for the calculation of $\mathsf{con}_{\mathsf{W}}(\alpha, c)$:*

Step 1: Find all addresses d for which $R_\alpha(c, d)$ holds. They form a set M.

Step 2: Output the maximum with respect to the lexicographic ordering over all words u such that $d : u$ occurs in α for $d \in M$. If $M = \varnothing$, output ϵ.

Output ε if α is not a RD, or c is not an address of α.

Lemma 15 *The function $\mathsf{con}_{\mathsf{W}}$ is polynomial time computable.*

Proof. It can be checked in polynomial time whether α is a RD containing address c. If not, $\mathsf{con}_{\mathsf{W}}(\alpha, c)$ is evaluated immediately as ε. If α is a RD, for each address c occurring in α the addresses d with $R_\alpha(c, d)$ can be found in polytime. The number of addresses occurring in α is bounded by the length of α [5]. This yields that the

[5] We use here natural assumptions about the function coding RDs as words.

set M can be constructed in polynomial time relative to α. Then, the required maximum can be found in polynomial time relative to M and α. $\square$

Example 16 *Let us calculate* $\mathsf{con}_{\mathsf{W}}(\alpha, n)$ *for* α *being the realisation dag presented in example 9. All addresses are reachable from* n*, therefore* M *is the set of all addresses of* α*.* $0 : \epsilon$ *occurs in* α *but no other RD parts of this form. Therefore* $\mathsf{con}_{\mathsf{W}}(\alpha, n) = \epsilon$*.*

Notations 17 *For a term* t*, let us write* $value(t)$ *for the word with* $\mathcal{TM} \vDash t = \overline{value(t)}$ *if there exists any.*

Lemma 18 *Let* α *be a RD, and* c *one of its addresses. Let* t *be a term. Then, the following holds.*

$$\mathsf{con}(\alpha, c)\ \mathfrak{R}\ t \in \mathsf{W} \Rightarrow \mathsf{con}_{\mathsf{W}}(\alpha, c) = value(t)$$

Proof. It follows from the definition of con that the set of reachable addresses from c contains exactly one address d with $v(d)$ being a leaf of α. Because of the assumption $v(d)$ has to be indexed by the value of t which yields the claim. $\square$

The function W_b, which depends on $\mathsf{con}_{\mathsf{W}}$, helps to bound realisation functions.

Definition 19 (W_b) *Let* b *be an address finder. The function* $\mathsf{W}_b : \mathbb{W} \times \mathbb{W} \to \mathbb{W}$ *is defined as*

$$\mathsf{W}_b(w, \alpha) := max\{\mathsf{con}_{\mathsf{W}}(\alpha, b(\alpha, v)) : \epsilon \subset v \subseteq w\},$$

where max refers to the lexicographic ordering of the words. W_b *outputs* ε *if* α *is not a RD.*

W_b is polytime because $\mathsf{con}_{\mathsf{W}}$ is polytime as well.

Example 20 *Let us explain the behaviour of the function* W_b *in a typical example. Let its second argument* α *have the following property:* $\alpha\ \mathfrak{r}_b\ A_1, \cdots, A_n$*. Then,* $\mathsf{W}_b(n, \alpha)$ *outputs the largest word* u *in a RD part* $c : u$ *of* α *for* c *reachable from an address in the set* $\{b(\alpha, 1), b(\alpha, 2), \cdots, b(\alpha, n)\}$*.*

The polynomial time computable function defined below is important to bound realisation functions too.

Definition 21 (Maximal address function) *The function* $\mathsf{MA} : \mathbb{W} \to \mathbb{W}$ *applied to a RD* α *returns its maximal address with respect to the lexicographic order. It returns* ε *if* α *is not a RD.*

Example 22 *Let us apply the function* MA *to* α *being the RD from example 9. Clearly, we have* $\mathsf{MA}(\alpha) = n$*.*

6. Applying the formalism to $\mathsf{T}^i_{\mathsf{PT}}$

From now on, we work with a sequent style formulation of $\mathsf{T}^i_{\mathsf{PT}}$ which we call $\mathsf{T}^i_{\mathsf{PT}}$ as well.

6.1. Stating the main claim

Notations 23 *The following notations allow to state the main theorem concisely.*

- Γ *is always a sequence of positive formulas of the form* $A_1, \ldots, A_n$. $|\Gamma|$ *gives its length* n.
- $\Gamma, A[\vec{s}]$ *denotes* $\Gamma[\vec{s}], A[\vec{s}]$.
- *We often use* $+$ *and* $\cdot$ *instead of* $*$ *and* $\times$. *In such contexts natural numbers* n *denote the word* $\underbrace{00\cdots0}_{n \text{ times}}$ *as usual. We sometimes write* $w-1$ *instead of* $\mathsf{p}_{\mathsf{W}}(w)$ *for* $w \in \mathbb{W}$.
- *As in example 20, the function* W_b *will always occur in connection with a sequence of formulas of length* n, *and we will always take* n *as its first argument. Therefore, we suppress it always.*
- *For a RD* α, $\alpha^{\ominus}$ *denotes the RD produced by deleting the RD parts of* α *which contain the maximal address of* α. $\alpha^{\ominus}$ *is defined to be the empty word if* α *is not a RD. Clearly, we have* $w^{\ominus} \leq w$ *for all words* w [6].

Theorem 24 *Let* Γ, D *be a sequence of positive formulas. Assume that there is a proof of* $\Gamma \Rightarrow D$ *in* $\mathsf{T}^i_{\mathsf{PT}}$ *that uses only positive formulas. Assume that a polytime address finder* b *is given. Then there exist polytime functions* $p^{-1}, \delta, \kappa, \gamma$ *(independent of* b*) and a polytime realisation function* p_b *such that for all* $\vec{s}$ *and for all* α *that are realisers of* $\Gamma[\vec{s}]$ *relative to* b *the following five properties hold:*

(1)
- $p^{-1}(p_b(\alpha)) = \alpha$.
- $p^{-1}(w) \leq w$ *for all* $w \in \mathbb{W}$.

(2) $p_b(\alpha)$ $\mathfrak{r}_{b^*}$ $\Gamma, D[\vec{s}]$, *where* b^* *is the following address finder.*

$$b^*(\rho, i) = \begin{cases} b(p^{-1}(\rho), i), & \text{if } 1 \leq i \leq |\Gamma| \\ \mathsf{MA}(\rho), & \text{if } i = |\Gamma| + 1 \\ \epsilon, & \text{else} \end{cases}$$

[6] Again, we use natural assumptions about the function coding RDs as words that is silently assumed.

(3) $\mathsf{MA}(p_b(\alpha)) \leq \mathsf{MA}(\alpha) + \kappa(\mathsf{W}_b(\alpha))$.

(4) $p_b(\alpha) \leq \alpha + \delta(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha))$.

(5) $\mathsf{W}_{b^*}(p_b(\alpha)) \leq \gamma(\mathsf{W}_b(\alpha))$.

(1) claims that we have an inverse function for the realisation function. The inverses can be defined because the realisation functions always add something to the given realiser (we will assume this tacitly in the whole realisation proof). The realisation functions will always store the new information under addresses which are not used yet. This guarantees that we construct again a RD.

(2) claims that the application of the realisation function to a realiser of $\Gamma[\vec{s}]$ delivers a realiser of $\Gamma, D[\vec{s}]$ such that the standard realisers of the formulas of $\Gamma[\vec{s}]$ are constructed from the same addresses as before. The standard realiser of $D[\vec{s}]$ is constructed from the maximal address.

All realisation functions we use apply the address finder only to relevant inputs. Therefore, we will tacitly assume that for two address finders b and b' that fulfil $b(w,i) = b'(w,i)$ for all relevant inputs i and all words w, the same realisation functions are produced. This allows us to define address finders only for relevant inputs in the following.

(3) claims that we can control the length of the maximal address. It is important that the bound depends only on $\mathsf{W}_b(\alpha)$ but not on $\mathsf{MA}(\alpha)$.

(4) and (5) make analogue statements for the whole realiser. The suppressed first arguments in (5) are $|\Gamma| + 1$ or $|\Gamma|$, respectively.

We will prove the main theorem by simultaneous induction on the depth of the positive proof of $\Gamma \Rightarrow D$ in $\mathsf{T}^i_{\mathsf{PT}}$. The bounding properties 3 and 4 will be needed to deal with induction, property 5 for cut. Because it increases legibility, we will always find first the p_b-functions, and only then construct the other polytime functions $(p^{-1}, \delta, \kappa, \gamma)$. This is legitimate because these functions will always be constructed independently of b or p_b.

6.2. Realisation functions for the axioms

Let us show that for proof depth 0 the claim holds. We illustrate some interesting or explanatory examples.

6.2.1. Equation axioms

Let us realise

$$\Gamma, s = t, s \in \mathsf{W}, t \in \mathsf{W} \Rightarrow \mathsf{d}_{\mathsf{W}}(p, q, s, t) = p.$$

Assume $\alpha\ \mathfrak{r}_b\ \Gamma, s = t, s \in \mathsf{W}, t \in \mathsf{W}[\vec{s}]$ for an address finder b [7]. This implies the existence of standard realisers for the main formulas relative to the substitution $[\vec{s}]$ and therefore $\mathcal{TM} \vDash \mathsf{d_W}(p, q, s, t) = p[\vec{s}]$. This means that we can realise the succedent trivially and get the realiser we searched by adding the RD part $\mathsf{MA}(\alpha) + 1 : \epsilon$ to α. We define p_b as

$$p_b(\rho) := \rho/\mathsf{MA}(\rho) + 1 : \epsilon.$$

A function that satisfies the requirements of the inverse is p^{-1}, defined as

$$p^{-1}(\rho) := \rho^{\ominus}.$$

Let us check that 2 holds. Because p^{-1} is the inverse of p_b, we have for $1 \leq i \leq |\Gamma|$

$$b^*(p_b(\alpha), i) = b(\alpha, i).$$

Because of the assumption about α, this yields for $1 \leq i \leq |\Gamma|$

$$\mathsf{con}\Big(p_b(\alpha), b^*\big[p_b(\alpha), i\big]\Big)\ \mathfrak{R}\ A_i[\vec{s}].$$

To show yet is

$$\mathsf{con}\Big(p_b(\alpha), b^*\big[p_b(\alpha), |\Gamma| + 1\big]\Big)\ \mathfrak{R}\ \mathsf{d_W}(p, q, s, t) = p[\vec{s}].$$

$b^*\big(p_b(\alpha), |\Gamma| + 1\big)$ is equal to $\mathsf{MA}(\alpha) + 1$. So $\mathsf{con}\Big(p_b(\alpha), b^*\big(p_b(\alpha), |\Gamma| + 1\big)\Big)$ is equal to ϵ. This delivers 2.

p_b increases the maximal address of its argument only by one and the length of the information added by p_b can be bounded polynomially in $\mathsf{MA}(\alpha)$. Therefore, 3 and 4 are satisfied. To see that 5 is satisfied, let us calculate $\mathsf{W}_{b^*}(p_b(\alpha))$, which is the maximum of the set

$$\{\mathsf{con_W}\Big(p_b(\alpha), b^*\big[p_b(\alpha), i\big]\Big) : 1 \leq i \leq |\Gamma| + 1\}.$$

$\mathsf{W}_b(\alpha)$ is the maximum of the set

$$\{\mathsf{con_W}\Big(\alpha, b\big[\alpha, i\big]\Big) : 1 \leq i \leq |\Gamma|\}.$$

Because of the definition of b^* and because p^{-1} is the inverse function of p_b, the two sets are identical except for the element

$$\mathsf{con_W}\Big(p_b(\alpha), b^*\big[p_b(\alpha), |\Gamma| + 1\big]\Big),$$

which equals ϵ. Therefore, the two maxima are the same. Other equation axioms can be realised analogously. We note that, given a correct inverse, to prove 2 and 5, we only have to check the content stored at the maximal address.

[7] Even if s and $\vec{s}$ look related, they are completely independent. Similarly for t.

6.2.2. Compositional truth

To realise these axioms the use of pointers will be crucial not to violate 3 or 4. We will construct the realisation function for the following axiom.

$$\Gamma, \mathsf{T}(s\dot{\vee}t) \Rightarrow \mathsf{T}(s) \vee \mathsf{T}(t)$$

Assume $\alpha \Vdash_b \Gamma, \mathsf{T}(s\dot{\vee}t)[\vec{s}]$ for an address finder b. We are interested in the realisation information for $\mathsf{T}(s\dot{\vee}t)[\vec{s}]$. Because $\mathsf{T}(s\dot{\vee}t)[\vec{s}]$ is realised exactly as $(\mathsf{T}(s) \vee \mathsf{T}(t))[\vec{s}]$, we only have to point to its address. We define p_b as

$$p_b(\rho) := \rho/\mathsf{MA}(\rho) + 1 \to b(\rho, |\Gamma| + 1).$$

A function p^{-1} that satisfies the requirements of the inverse can be defined as

$$p^{-1}(\rho) := \rho^{\ominus}.$$

By similar reasoning as before, one can show that properties 1 until 5 are satisfied. Observe that conditions 3 and 4 might be violated if we would just reproduce the realisation information stored at $b(\rho, |\Gamma| + 1)$ instead of using an additional edge.

Let us look now at the axiom

$$\Gamma, s \in \mathsf{W}, \mathsf{T}(\dot{\mathsf{W}}st) \Rightarrow t \in \mathsf{W}.$$

We use an auxiliary function con_T defined as follows.

Definition 25 (con_T) *The function $\mathsf{con}_\mathsf{T} : \mathbb{W} \times \mathbb{W} \to \mathbb{W}$ is defined as con_W with the only difference that it outputs the maximum over all words u such that $d : \langle \ulcorner \mathsf{T} \urcorner, u \rangle$ is a part of α for $d \in M$. If M is empty, it also outputs ϵ.*

The function con_T is polytime for the same reasons as con_W. Using con_T, we define the realisation function p_b as follows.

$$p_b(\rho) := \rho/\mathsf{MA}(\rho) + 1 : \mathsf{con}_\mathsf{T}(\rho, |\Gamma| + 2)$$

The realisation information of the formula $s \in \mathsf{W}$ does not occur in the realisation function, nevertheless the bound s for t is needed. Let us explain why.

The added realisation information for $t \in \mathsf{W}$ could increase the maximum of the computational content calculated by W_b. Indeed, the value of t is already present in the realiser of the antecedent. But the function con_W that extracts computational content ignores RD parts of the form $c : \langle \ulcorner \mathsf{T} \urcorner, w \rangle$. Therefore, only the presence of the realisation information for $s \in \mathsf{W}$ assures that conditions 4 and 5 are not violated in this case. This shows where our approach would fail for truth theories of the strength PRA containing the additional axiom

$$\Gamma, \mathsf{T}(\dot{\mathsf{W}}t) \Rightarrow t \in \mathsf{W}.$$

6.3. Realisation functions for the conclusions of rules

We illustrate some interesting or difficult examples. We leave away the $\vee$-right - and the quantifier rules because they can be realised easily. The $\wedge$-right-rule is realised similarly as cut.

6.3.1. $\vee$-left rule

Let the applied $\vee$-left rule have the following form.

$$\frac{\Gamma, A \Rightarrow D \qquad \Gamma, B \Rightarrow D}{\Gamma, A \vee B \Rightarrow D}$$

By induction hypothesis, we have realisation functions p and q for both premises. Assume $\alpha \; \mathfrak{r}_b \; \Gamma, (A \vee B)[\vec{s}]$ for an address finder b. We have to make a distinction by cases according to the disjunct of $(A \vee B)[\vec{s}]$ which is realised by α. Depending on this, we will apply p or q. To the result of this application we add a marker which tells us which function has been applied. This allows the definition of an inverse function which works for both cases.

Let us now give the realisation function for an arbitrary input $\rho \in \mathbb{W}$ in detail. We will modify the input ρ before applying p or q since they expect a realiser of $\Gamma, A[\vec{s}]$ or $\Gamma, B[\vec{s}]$, respectively. This is done using the following auxiliary function.

Definition 26 ($\downarrow$) *The function $\downarrow\colon \mathbb{W} \times \mathbb{W} \to \mathbb{W}$ applied to a RD α and an address c of α returns the address of the vertex at the end of a maximal $\to$-path p in α starting at $v(c)$ containing only non-indexed edges except of possibly its last edge. We write $c\downarrow$ for $\downarrow(\alpha, c)$ if α is clear from the context. $\downarrow$ outputs ε if one of the inputs is not as intended.*

The modified RD is $\rho/\mathsf{MA}(\rho) + 1 \to b(\rho, |\Gamma| + 1)\downarrow$, which we abbreviate as ρ'.

We find the realisation information contained in ρ' by the following address finder b'.

$$b'(\rho, i) := \begin{cases} b(\rho^{\ominus}, i), & \text{if } 1 \leq i \leq |\Gamma| \\ \mathsf{MA}(\rho), & \text{if } i = |\Gamma| + 1 \end{cases}$$

We define a second auxiliary function h.

Definition 27 (h) *For a RD α and an address c of α, let the path p be given as in the last definition. Then, the function $h : \mathbb{W} \times \mathbb{W} \to \mathbb{W}$ outputs 0 if p contains an edge indexed by 0 and 1 else. h outputs ε if one of the inputs is not as intended.*

We define the property C to hold, exactly if $h(\rho, b(\rho, |\Gamma|+1)) = 0$. Note that if ρ is given as intended, C holds exactly if ρ' is a realiser of $\Gamma, A[\vec{s}]$. Now, we can define f_b as follows.

$$f_b(\rho) := \begin{cases} p_{b'}(\rho')/\mathsf{MA}\big[p_{b'}(\rho')\big] + 1 : 0/\mathsf{MA}\big[p_{b'}(\rho')\big] + 2 \to \mathsf{MA}\big[p_{b'}(\rho')\big], & \text{if } C \\ q_{b'}(\rho')/\mathsf{MA}\big[q_{b'}(\rho')\big] + 1 : 1/\mathsf{MA}\big[q_{b'}(\rho')\big] + 2 \to \mathsf{MA}\big[q_{b'}(\rho')\big], & \text{else} \end{cases}$$

The marker, stored in the second largest address tells us whether p or q was applied. Accordingly, we define f^{-1} as follows.

$$f^{-1}(\rho) := \begin{cases} p^{-1}(\rho^{\ominus\ominus})^{\ominus}, & \text{if } \mathsf{con}_{\mathsf{W}}(\rho, \mathsf{MA}(\rho) - 1) = 0 \\ q^{-1}(\rho^{\ominus\ominus})^{\ominus}, & \text{else} \end{cases}$$

We have to show that this function works as an inverse of f_b when f_b is applied to a realiser α of $\Gamma, (A \vee B)[\vec{s}]$ relative to b. First, we assume that α realises the first disjunct of $(A \vee B)[\vec{s}]$. The definition of the realisation relation delivers

$$\alpha' \mathrel{\mathfrak{r}_{b'}} \Gamma, A[\vec{s}].$$

Therefore, the induction hypothesis delivers $p^{-1}(p_{b'}(\alpha')) = \alpha'$. Similarly, if α realises the second disjunct, we have $q^{-1}(q_{b'}(\alpha_1)) = \alpha_1$. Altogether, this immediately implies property 1.

Let us show that property 2 holds. Again we assume that α realises the first disjunct of $(A \vee B)[\vec{s}]$, the other case works similarly.

$$\alpha' \mathrel{\mathfrak{r}_{b'}} \Gamma, A[\vec{s}]$$

implies because of the induction hypothesis for p

$$\mathsf{con}\Big(p_{b'}(\alpha'), \mathsf{MA}\big[p_{b'}(\alpha')\big]\Big) \mathrel{\mathfrak{R}} D[\vec{s}],$$

which yields property 2 because of the correctness of the inverse. Now, we prove property 3. Let us again assume that α realises the first disjunct of $(A \vee B)[\vec{s}]$, the other case works similarly. The induction hypothesis delivers

$$\mathsf{MA}(f_b(\alpha)) \leq \mathsf{MA}(\alpha') + \kappa_p(\mathsf{W}_{b'}(\alpha')) + 2.$$

(2 corresponds to the marker and the added copy.) Clearly, we have $\mathsf{W}_{b'}(\alpha') = \mathsf{W}_b(\alpha)$ and $\mathsf{MA}(\alpha') = \mathsf{MA}(\alpha) + 1$. Therefore, we get

$$\mathsf{MA}(f_b(\alpha)) \leq (\mathsf{MA}(\alpha) + 1) + \kappa_p(\mathsf{W}_b(\alpha)) + 2.$$

For the other case, the same bounding polynomial but with κ_p replaced by κ_q could be found. Therefore, for a polynomial bounding κ_p and κ_q property 3 is fulfilled. Property 4 can be proved similarly. Property 5 follows easily from $\mathsf{W}_b(\alpha) = \mathsf{W}_{b'}(\alpha')$ and the induction hypothesis for p and q.

6.3.2. Cut

Let the applied cut rule have the following form.

$$\frac{\Gamma \Rightarrow A \qquad \Gamma, A \Rightarrow D}{\Gamma \Rightarrow D}$$

By induction hypothesis we have realisation functions p and q for the premises. Assume $\alpha \ \mathfrak{r}_b \ \Gamma[\vec{s}]$ for an address finder b. We define the new realisation function as composition of p and q. First, we apply p_b to get a realiser of $\Gamma, A[\vec{s}]$ relative to a b'. Then apply $q_{b'}$ to get a realiser of $\Gamma, A, D[\vec{s}]$. This is the realiser we need relative to an address finder that just forgets the address that contains the realisation information for $A[\vec{s}]$. We define b' as follows.

$$b'(\rho, i) := \begin{cases} b(p^{-1}(\rho), i), & \text{if } 1 \leq i \leq |\Gamma| \\ \mathsf{MA}(\rho), & \text{if } i = |\Gamma| + 1 \end{cases}$$

We define f_b as

$$f_b(\rho) := q_{b'}(p_b(\rho)).$$

We define f^{-1} as

$$f^{-1}(\rho) := p^{-1}(q^{-1}(\rho)).$$

We have to show that this function works as an inverse of f_b when f_b is applied to a realiser α of $\Gamma[\vec{s}]$ relative to b. From the induction hypothesis 2 for p we get

$$p_b(\alpha) \ \mathfrak{r}_{b'} \ \Gamma, A[\vec{s}]. \tag{A}$$

Now, the induction hypothesis 1 for q delivers $q^{-1}(q_{b'}[p_b(\alpha)]) = p_b(\alpha)$. Therefore, the induction hypothesis 1 for p delivers property 1.

From (A), we get by induction hypothesis for 2

$$\mathsf{con}(q_{b'}\big[p_b(\alpha)\big], \mathsf{MA}(q_{b'}\big[p_b(\alpha)\big]) \ \mathfrak{R} \ D[\vec{s}],$$

which implies property 2.

Let us prove now property 3. Because of the induction hypothesis for 5, we have $\mathsf{W}_{b'}(p_b(\alpha)) \leq \gamma_p(\mathsf{W}_b(\alpha))$. Using induction hypothesis 3, we have additionally

$$\begin{aligned} \mathsf{MA}(f_b(\alpha)) &\leq \mathsf{MA}(p_b(\alpha)) + \kappa_q(\mathsf{W}_{b'}(p_b(\alpha))) \leq \mathsf{MA}(\alpha) + \kappa_p(\mathsf{W}_b(\alpha)) + \kappa_q(\mathsf{W}_{b'}(p_b(\alpha))) \\ &\leq \mathsf{MA}(\alpha) + \kappa_p(\mathsf{W}_b(\alpha)) + \kappa_q(\gamma_p(\mathsf{W}_b(\alpha)))). \end{aligned}$$

Property 4 can be proved similarly.

Let us show now property 5. By induction hypothesis 5, the following two inequations hold.

$$\mathsf{W}_{b'}(p_b(\alpha)) \leq \gamma_p(\mathsf{W}_b(\alpha))$$

$$\mathsf{con}_{\mathsf{W}}(q_{b'}(p_b(\alpha)), \mathsf{MA}(q_{b'}(p_b(\alpha)))) \leq \gamma_q(\mathsf{W}_{b'}(p_b(\alpha)))$$

Therefore, we have for the composition $\gamma_q \circ \gamma_p$

$$\mathsf{W}_{b^*}(q_{b'}(p_b(\alpha))) \leq (\gamma_q \circ \gamma_p)(\mathsf{W}_b(\alpha)).$$

6.3.3. Induction

Let the applied induction rule have the following form.

$$\frac{\Gamma \Rightarrow \mathsf{T}(r\epsilon) \qquad \Gamma, \mathsf{T}(rx), x \in \mathsf{W} \Rightarrow \mathsf{T}(r(\mathsf{s}_i x))}{\Gamma, t \in \mathsf{W} \Rightarrow \mathsf{T}(rt)}$$

By induction hypothesis we have realisation functions p, q_0 and q_1 for the premises.

As usual, we use recursion to define the realisation function. The main obstacle is to deliver the necessary bound, which will be produced using induction hypotheses 3 and 4.

The recursion works roughly in the following way: Given a realiser α of $\Gamma, t \in \mathsf{W}[\vec{s}]$ relative to b, we get by applying p_b to α a realiser of $\Gamma, \mathsf{T}(r\epsilon)[\vec{s}]$ relative to a b_1. When we add to $p_b(\alpha)$ a suitable RD part, we get a realiser of $\Gamma, \mathsf{T}(r\epsilon), \epsilon \in \mathsf{W}[\vec{s}]$ relative to a b_2. We can apply the functions $(q_0)_{b_2}$ or $(q_1)_{b_2}$ to get a realiser of $\Gamma, \mathsf{T}(r\overline{0})[\vec{s}]$ or $\Gamma, \mathsf{T}(r\overline{1})[\vec{s}]$ relative to a b_3. Then again, by adding a suitable RD part, we get a realiser of e.g. $\Gamma, \mathsf{T}(r\overline{0}), \overline{0} \in \mathsf{W}[\vec{s}]$ relative to a b_4 and can apply the functions $(q_0)_{b_4}$ or $(q_1)_{b_4}$ to get a realiser of e.g. $\Gamma, \mathsf{T}(r\overline{00})[\vec{s}]$. This process can be iterated arbitrary often and will deliver after $|value(t[\vec{s}])|$ many iterations the searched realiser.

Nevertheless, two problems have to be solved yet:

1. We have to use always the same recursion step functions. Therefore, we need an address finder $\tilde{b}$ such that for each $w \in \mathbb{W}$, after $|w|$ many recursion steps we still have a realiser of $\Gamma, \mathsf{T}(r\overline{w}), \overline{w} \in \mathsf{W}[\vec{s}]$ relative to $\tilde{b}$.

2. We have to deliver a bound for the sketched recursion.

Our strategy is to define first a binary function f. Its first argument is considered to be a realiser of $\Gamma, t \in \mathsf{W}[\vec{s}]$, the length of the second argument gives the number of iterations of the above described process to be carried out. Later, from this binary function, we easily define the realisation function.

Let us tackle now the first problem for the above sketched binary function. The $(q_i)_{\tilde{b}}$ which we will apply in the recursion step always ask for the realisation information

for $\Gamma[\vec{s}]$, which is stored in the first argument of the function. Therefore $\tilde{b}$ relies on an inverse of f which we define below.

Definition 28 *The function $f^{-1} : \mathbb{W} \times \mathbb{W} \to \mathbb{W}$ is defined by recursion as follows.*

$$\begin{array}{rcl} f^{-1}(\rho, \epsilon) & := & p^{-1}(\rho^{\ominus}) \\ f^{-1}(\rho, \mathsf{s}_i w) & := & f^{-1}(q_i^{-1}(\rho^{\ominus}), w) \end{array}$$

This function is clearly polynomial time computable since it can be given by a recursion bounded by ρ. We define $\tilde{b}$ which is the function mentioned in the first problem mentioned above.

Definition 29 *Assume that b is an address finder. The function $\tilde{b} : \mathbb{W} \times \mathbb{W} \to \mathbb{W}$ is given by the following definition of cases.*

$$\tilde{b}(\rho, i) = \begin{cases} b\Big(f^{-1}\big[\rho, \mathsf{con}_{\mathsf{W}}(\rho, \mathsf{MA}(\rho))\big], i\Big), & \text{if } 1 \leq i \leq |\Gamma| \\ \mathsf{MA}(\rho) - 1, & \text{if } i = |\Gamma| + 1 \\ \mathsf{MA}(\rho), & \text{if } i = |\Gamma| + 2 \end{cases}$$

Using $\tilde{b}$ the earlier mentioned function f_b can be defined.

Definition 30 *The function $f_b : \mathbb{W} \times \mathbb{W} \to \mathbb{W}$ is defined by recursion as follows.*

$$\begin{array}{rcl} f_b(\rho, \epsilon) & := & p_b(\rho)/\mathsf{MA}(p_b(\rho)) + 1 : \epsilon \\ f_b(\rho, \mathsf{s}_i w) & := & (q_i)_{\tilde{b}}(f_b(\rho, w))/\mathsf{MA}\Big((q_i)_{\tilde{b}}\big(f_b(\rho, w)\big)\Big) + 1 : \mathsf{s}_i w \end{array}$$

Example 31 *Let us give concrete examples for the above defined functions. We look another time at the function r which was defined at page 9 and the sequent*

$$x \in \mathsf{W} \Rightarrow \mathsf{T}(rx),$$

which cannot be realised by a polytime function using the standard realisation approach. It can be derived by the following induction.

$$\frac{\Rightarrow \mathsf{T}(r\epsilon) \qquad \mathsf{T}(rx), x \in \mathsf{W} \Rightarrow \mathsf{T}(r(\mathsf{s}_i x))}{t \in \mathsf{W} \Rightarrow \mathsf{T}(rt)}$$

So, if we deliver realisation functions for the premises, we can use the above defined functions to construct a realisation function f for the conclusion. We will construct f_{Id}, for Id defined as the function $\lambda xy.y$ on words, using premise realisation functions, but note that all introduced functions are independent of address finders

since there are no side formulas. We will use a realisation function p_{Id} for the first premise, e.g.

$$p_{Id}(\rho) := \rho/\mathsf{MA}(\rho) + 1 : \epsilon.$$

We also use the realisation functions $(q_i)_{\tilde{Id}}$ for the induction step premises, where $\tilde{Id}$ is the following function (note that Id has no relevant inputs).

$$\tilde{Id}(\rho, i) = \begin{cases} \mathsf{MA}(\rho) - 1, & \text{if } i = 1 \\ \mathsf{MA}(\rho), & \text{if } i = 2 \end{cases}$$

Realisation functions $(q_i)_{\tilde{Id}}$ for the induction step can be given as

$$(q_i)_{\tilde{Id}}(\rho) := \rho/\mathsf{MA}(\rho) + 1 \xrightarrow{0} \mathsf{MA}(\rho) - 1/\mathsf{MA}(\rho) + 1 \xrightarrow{1} \mathsf{MA}(\rho) - 1.$$

Let us now calculate $f_{Id}(\rho, w)$ for $\rho, w \in \mathbb{W}$ with f defined as in definition 30. We get

- $f_{Id}(\rho, \epsilon) = \rho/\mathsf{MA}(\rho) + 1 : \epsilon/\mathsf{MA}(\rho) + 2 : \epsilon$
- $f_{Id}(\rho, 0) = \rho/\mathsf{MA}(\rho)+1 : \epsilon/\mathsf{MA}(\rho)+2 : \epsilon/\mathsf{MA}(\rho)+3 \xrightarrow{0} \mathsf{MA}(\rho)+1/\mathsf{MA}(\rho)+3 \xrightarrow{1} \mathsf{MA}(\rho) + 1/\mathsf{MA}(\rho) + 4 : 0$
- $f_{Id}(\rho, 00) = \rho/\mathsf{MA}(\rho)+1 : \epsilon/\mathsf{MA}(\rho)+2 : \epsilon/\mathsf{MA}(\rho)+3 \xrightarrow{0} \mathsf{MA}(\rho)+1/\mathsf{MA}(\rho)+3 \xrightarrow{1} \mathsf{MA}(\rho) + 1/\mathsf{MA}(\rho) + 4 : 0/\mathsf{MA}(\rho) + 5 \xrightarrow{0} \mathsf{MA}(\rho) + 3/\mathsf{MA}(\rho) + 5 \xrightarrow{1} \mathsf{MA}(\rho) + 3/\mathsf{MA}(\rho) + 6 : 00$
- $\cdots$
- $f_{Id}(\rho, n) = \rho/\mathsf{MA}(\rho)+1 : \epsilon/\mathsf{MA}(\rho)+2 : \epsilon/\mathsf{MA}(\rho)+3 \xrightarrow{0} \mathsf{MA}(\rho)+1/\mathsf{MA}(\rho)+3 \xrightarrow{1} \mathsf{MA}(\rho) + 1/\mathsf{MA}(\rho) + 4 : 0/\mathsf{MA}(\rho) + 5 \xrightarrow{0} \mathsf{MA}(\rho) + 3/\mathsf{MA}(\rho) + 5 \xrightarrow{1} \mathsf{MA}(\rho) + 3/\mathsf{MA}(\rho)+6 : 00/\cdots/\mathsf{MA}(\rho)+(2n+1) \xrightarrow{0} \mathsf{MA}(\rho)+(2n-1)/\mathsf{MA}(\rho)+(2n+1) \xrightarrow{1} \mathsf{MA}(\rho) + (2n-1)/\mathsf{MA}(\rho) + (2n+2) : n$

(Analogously for arbitrary words of the same length as second argument.) It can be easily seen that $f_{Id}(\rho, w)$ is a realiser of $\mathsf{T}(r\overline{w}), \overline{w} \in \mathsf{W}$ relative to $\tilde{Id}$ for any $\rho, w \in \mathbb{W}$. The function f_{Id} is polytime because of its small growth. How do we get from f_{Id} a realisation function $\mathfrak{f}_b$ for the sequent $t \in \mathsf{W} \Rightarrow \mathsf{T}(rt)$? The realisation information for $t \in \mathsf{W}[\vec{s}]$ tells us how many and which recursion steps have to take place which delivers the second argument for f_{Id}. Therefore, we get a realisation function $\mathfrak{f}_b$ for the sequent as

$$\mathfrak{f}_b(\rho) := f_{Id}\Big(\rho, \mathsf{con}_{\mathsf{W}}\big[\rho, b(\rho, 1)\big]\Big).$$

To put the realiser of the formula $\mathsf{T}(rt)$ *to the last position, we use a copy.*

In the following, we will show how to find the realisation function $\mathfrak{f}_b$ *for arbitrary conclusions of the induction rule. The additional difficulty is that in general the function* f_b *is not polytime. Usually, we have to control the recursion with a bound.*

The next lemma claims the correctness of the function f from definition 30 and of its inverse f^{-1} from definition 28.

Lemma 32 *Let* α *be a realiser of* $\Gamma, t \in \mathsf{W}[\vec{s}]$ *relative to* b*. Then for each* $w \in \mathbb{W}$ *(A) and (B) hold.*

(A) $f_b(\alpha, w)\ \mathfrak{r}_{\tilde{b}}\ \Gamma, \mathsf{T}(r\overline{w}), \overline{w} \in \mathsf{W}[\vec{s}]$

(B) $f^{-1}(f_b(\alpha, w), w) = \alpha$

Proof. We show (A) and (B) by simultaneous induction on w. If w equals ϵ, both claims follow immediately from properties 1 and 2 for p.

Let us switch to an $\mathsf{s}_i w \in \mathbb{W}$. The induction hypothesis for (A) delivers

$$f_b(\alpha, w)\ \mathfrak{r}_{\tilde{b}}\ \Gamma, \mathsf{T}(r\overline{w}), \overline{w} \in \mathsf{W}[\vec{s}].$$

Therefore property 1 for q_i implies

$$(q_i)^{-1}(f_b(\alpha, \mathsf{s}_i w)^{\ominus}) = f_b(\alpha, w).$$

Together with the induction hypothesis for (B), this delivers (B) for $\mathsf{s}_i w$.

Property 2 of q_i and the induction hypothesis for (A) imply that the maximal address of $(q_i)_{\tilde{b}}(f_b(\alpha, w))$ contains the realisation information for $\mathsf{T}(r(\mathsf{s}_i\overline{w}))$. It follows that the second largest - and largest address of $f_b(\alpha, \mathsf{s}_i w)$ contain the realisation information for $\mathsf{T}(r(\mathsf{s}_i\overline{w}))$ and $\mathsf{s}_i\overline{w} \in \mathsf{W}$, respectively. Together with these facts, (B) for $\mathsf{s}_i w$ implies

$$f_b(\alpha, \mathsf{s}_i w)\ \mathfrak{r}_{\tilde{b}}\ \Gamma, \mathsf{T}(r(\mathsf{s}_i\overline{w})), \mathsf{s}_i\overline{w} \in \mathsf{W}[\vec{s}],$$

which finishes the proof. □

To bound the function f_b by a polynomial for first arguments that realise $\Gamma, t \in \mathsf{W}[\vec{s}]$, it will be necessary to bound the values of $\mathsf{W}_{\tilde{b}}(f_b(\alpha, w))$ for $w \in \mathbb{W}$. This is so, because the length of the added parts in each recursion step of f_b depends polynomially on $\mathsf{W}_{\tilde{b}}(f_b(\alpha, w))$ for a certain $w \in \mathbb{W}$.

Lemma 33 *Let* α *be a realiser of* $\Gamma, t \in \mathsf{W}[\vec{s}]$ *relative to* b *and let* $w \in \mathbb{W}$ *be less or equal value*$(t[\vec{s}])$*. Then we have*

$$\mathsf{W}_{\tilde{b}}(f_b(\alpha, w)) \leq \mathsf{W}_b(\alpha)$$

Proof. Let us calculate $\mathsf{W}_{\tilde{b}}(f_b(\alpha, w))$. Because of lemma 32, we have for $1 \leq i \leq |\Gamma|$

$$\tilde{b}(f_b(\alpha, w), i) = b(\alpha, i).$$

Therefore, the content at these addresses does not violate the inequation. Let us look at the $|\Gamma| + 1$-th relevant address. Because of lemma 32, we have

$$\mathsf{con}\Big(f_b(\alpha, w), \tilde{b}\big(f_b(\alpha, w), |\Gamma| + 1\big)\Big) \mathrel{\mathfrak{R}} \mathsf{T}(r\overline{w}).$$

Because of the stipulation that RD parts of the form $c : \langle \ulcorner \mathsf{T} \urcorner, v \rangle$ do not contribute to the computational content, we have

$$\mathsf{con}_{\mathsf{W}}\Big(f_b(\alpha, w), \tilde{b}\big(f_b(\alpha, w), |\Gamma + 1|\big)\Big) = \epsilon.$$

Because we have $w \leq \mathit{value}(t[\vec{s}])$ also the realisation information stored at the $|\Gamma| + 2$-th relevant address does not violate the inequation. □

The lemma we just proved allows to find bounding polynomials for $f_b(\alpha, w)$ and $\mathsf{MA}(f_b(\alpha, w))$ for suitably chosen α and w.

Lemma 34 *There is a polynomial $\kappa_f : \mathbb{W} \to \mathbb{W}$ such that for all address finders b, all $\vec{s}$, all realisers α of $\Gamma, t \in \mathsf{W}[\vec{s}]$ relative to b, and all $w \leq \mathit{value}(t[\vec{s}])$, we have*

$$\mathsf{MA}(f_b(\alpha, w)) \leq \mathsf{MA}(\alpha) + \kappa_f(\mathsf{W}_b(\alpha)).$$

Proof. Because property 3 holds for p, q_0, q_1, we have MA-bounding polynomials $\kappa_p, \kappa_{q_0}, \kappa_{q_1}$. Let κ_q be a polynomial that bounds κ_{q_0} and κ_{q_1}. Using the properties of the bounding functions, we derive

$$\mathsf{MA}(f_b(\alpha, w)) \leq \mathsf{MA}(\alpha) + \kappa_p(\mathsf{W}_b(\alpha)) + 1 + \sum_{v \subset w} \Big(\kappa_q\big[\mathsf{W}_{\tilde{b}}(f_b(\alpha, v))\big] + 1\Big).$$

Using lemma 33, we get

$$\mathsf{MA}(f_b(\alpha, w)) \leq \mathsf{MA}(\alpha) + \kappa_p(\mathsf{W}_b(\alpha)) + 1 + \kappa_q(\mathsf{W}_b(\alpha)) \cdot w + w.$$

This implies our claim because we have $w \leq \mathsf{W}_b(\alpha)$. □

Lemma 35 *There is a polynomial $\delta_f : \mathbb{W} \to \mathbb{W}$ such that for all address finders b, all $\vec{s}$, all realisers α of $\Gamma, t \in \mathsf{W}[\vec{s}]$ relative to b, and all $w \leq \mathit{value}(t[\vec{s}])$, we have*

$$f_b(\alpha, w) \leq \alpha + \delta_f(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha)).$$

Proof. Because property 4 holds for p, q_0, q_1 by induction hypothesis, we have bounding polynomials $\delta_p, \delta_{q_0}, \delta_{q_1}$. Let δ_q be a polynomial that bounds δ_{q_0} and δ_{q_1}. The RD parts of the form $c : w$ we add in the course of the recursion after using an induction premise function can be bounded by a polynomial h in $\mathsf{MA}(\alpha)$ and $\mathsf{W}_b(\alpha)$ because of lemma 34. Altogether, this implies

$$f_b(\alpha, w) \leq \alpha + \delta_p\big(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha)\big) + h\big(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha)\big) + \sum_{v \subset w} \left(\delta_q\Big(\mathsf{W}_{\tilde{b}}\big[f_b(\alpha, v)\big], \mathsf{MA}\big[f_b(\alpha, v)\big]\Big) + h\big[\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha)\big] \right).$$

Using lemmas 33 and 34 we derive

$$f_b(\alpha, w) \leq \alpha + \delta_p\big(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha)\big) + h\big(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha)\big) + \sum_{v \subset w} \left(\delta_q\Big(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha) + \kappa_f(\mathsf{W}_b(\alpha))\Big) + h\big[\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha)\big] \right).$$

The summands are not dependent on the sum variable v, so we get

$$f_b(\alpha, w) \leq \alpha + \delta_p\big(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha)\big) + h\big(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha)\big) + w \cdot \left(\delta_q\Big(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha) + \kappa_f(\mathsf{W}_b(\alpha))\Big) + h\big[\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha)\big] \right).$$

This implies our claim because we have $w \leq \mathsf{W}_b(\alpha)$. $\square$

Now, using the binary function f_b, we can define the realisation function $\mathfrak{f}_b$ in the following way. First, we define a polytime variant $\hat{f}_b(\rho, v)$ of $f_b(\rho, v)$ by bounded recursion with bound $\rho + \delta_f(\mathsf{W}_b(\rho), \mathsf{MA}(\rho))$. Because of the previous lemma, $\hat{f}_b(\rho, v)$ equals $f_b(\rho, v)$ if ρ is a realiser of $\Gamma, t \in \mathsf{W}[\vec{s}]$ relative to b and v smaller or equal $value(t[\vec{s}])$.

To get a unary realisation function, we use realisation information for the formula $t \in \mathsf{W}[\vec{s}]$ stored in ρ to determine $value(t[\vec{s}])$. This is the missing second argument of $\hat{f}_b$. Therefore, we define the unary $h_b(\rho)$ as

$$\hat{f}_b\Big(\rho, \mathsf{con}_{\mathsf{W}}\big[\rho, b(\rho, |\Gamma| + 1)\big]\Big).$$

h_b delivers the realisation information for $\mathsf{T}(rt)[\vec{s}]$, but not under the maximal address. Therefore, we use a copy and define the realisation function $\mathfrak{f}_b$ as

$$\mathfrak{f}_b(\rho) := h_b(\rho)/\mathsf{MA}(h_b(\rho)) + 1 \to \mathsf{MA}(h_b(\rho)) - 1.$$

It can be seen immediately that all components of the function $\mathfrak{f}_b$ are polytime. Therefore, the following holds.

Lemma 36 *The function $\mathfrak{f}_b$ is polytime for any address finder b.*

To finish the proof of the main claim, we have to show that properties 1 until 5 hold for $\mathfrak{f}_b$.

For property 1, we have to define an inverse function for $\mathfrak{f}_b$ which must be correct for realiser inputs. For a realiser α of $\Gamma, t \in \mathsf{W}[\vec{s}]$ relative to b, we have because of lemma 35

$$\hat{f}_b\Big(\alpha, \mathsf{con}_{\mathsf{W}}\big[\alpha, b(\alpha, |\Gamma|+1)\big]\Big) = f_b\Big(\alpha, \mathsf{con}_{\mathsf{W}}\big[\alpha, b(\alpha, |\Gamma|+1)\big]\Big).$$

Therefore, using lemma 32, we get a correct inverse $\mathfrak{f}^{-1}$ defined as follows.

$$\mathfrak{f}^{-1}(\rho) = f^{-1}\Big(\rho^{\ominus}, \mathsf{con}_{\mathsf{W}}\big[\rho^{\ominus}, \mathsf{MA}(\rho^{\ominus})\big]\Big)$$

Lemmas 32 and 35 imply property 2. Property 3 follows from 34. Property 4 follows immediately from the definition of $\mathfrak{f}_b$. Property 5 follows because the formula which is realised additionally is a T-formula. This concludes the proof of the main theorem 24. The feasibility of $\mathsf{T}^i_{\mathsf{PT}}$ follows now as a corollary.

Corollary 37 (of theorem 24) *The provably total functions of $\mathsf{T}^i_{\mathsf{PT}}$ are exactly the polynomial time computable functions.*

Proof. The lower bound of T_{PT} follows from [12] as mentioned in the introduction.

Assume that the function $F : \mathbb{W} \to \mathbb{W}$ is provably total in $\mathsf{T}^i_{\mathsf{PT}}$ [8]. Therefore, for a corresponding closed t_F, we have

$$\mathsf{T}^i_{\mathsf{PT}} \vdash x \in \mathsf{W} \Rightarrow t_F x \in \mathsf{W}$$

By cut elimination we have a proof of this sequent only containing positive formulas. We can apply the main theorem 24 and get a polytime function f with properties 1 until 5. For an arbitrary $w \in \mathbb{W}$ we have for the identity address finder Id

$$0 : w \; \mathfrak{r}_{Id} \; \overline{w} \in \mathsf{W}.$$

Property 2 of f_{Id} delivers

$$\mathsf{con}\Big(f_{Id}(0:w), \mathsf{MA}\big[f_{Id}(0:w)\big]\Big) \; \mathfrak{R} \; t_F\overline{w} \in \mathsf{W},$$

which implies because of lemma 18

$$\mathsf{con}_{\mathsf{W}}\Big(f_{Id}(0:w), \mathsf{MA}\big[f_{Id}(0:w)\big]\Big) = value(t_F\overline{w}).$$

[8] The proof is easily adapted to functions with higher arity.

This implies

$$\mathsf{con}_{\mathsf{W}}\Big(f_{Id}(0:w), \mathsf{MA}\big[f_{Id}(0:w)\big]\Big) = F(w),$$

for all w in $\mathbb{W}$. Therefore, F is a polytime function.

□

6.4. Applying the formalism to (the classical) T_{PT}

To deal with classical logic, the new realisation formalism can be modified in exactly the same way as in Strahm [31]. The realisation functions always delivers a pair as output, where its first element determines, which formula D of the consequent is realised, and the second is a realiser of Γ, D. We use the following conventions which allow to state the new main theorem very similarly as before.

- For any function F whose image contains exclusively pairs, let f denote the function $\lambda x.F(x)_1$, where $F(x)_1$ is the second projection of $F(x)$.
- D_j always denotes the j-th formula of a sequence Δ of formulas.

Theorem 38 *Let Γ, Δ be a sequence of positive formulas. Assume that there is a T_{PT} proof of $\Gamma \Rightarrow \Delta$ that uses only positive formulas. Assume that a polytime address finder b is given. Then there exist polytime functions $p^{-1}, \delta, \kappa, \gamma$ (independent of b) and a polytime realisation function P_b such that for all $\vec{s}$ and for all α that are realisers of $\Gamma[\vec{s}]$ relative to b the following five properties hold:*

(1)
- *$p^{-1}(w) \leq w$ for all $w \in \mathbb{W}$.*
- *$p^{-1}(p_b(\alpha)) = \alpha$.*

(2) $p_b(\alpha)\ \mathfrak{r}_{b^*}\ \Gamma, D_j[\vec{s}]$ holds, where $P_b(\alpha)_0 = j$, and where b^* is the following function.

$$b^*(\rho, i) = \begin{cases} b(p^{-1}(\rho), i), & \text{if } 1 \leq i \leq |\Gamma| \\ \mathsf{MA}(\rho), & \text{if } i = |\Gamma| + 1 \end{cases}$$

(3) $\mathsf{MA}(p_b(\alpha)) \leq \mathsf{MA}(\alpha) + \kappa(\mathsf{W}_b(\alpha))$.

(4) $p_b(\alpha) \leq \alpha + \delta(\mathsf{W}_b(\alpha), \mathsf{MA}(\alpha))$.

(5) $\mathsf{W}_{b^*}(p_b(\alpha)) \leq \gamma(\mathsf{W}_b(\alpha))$.

This main theorem can again be proved by induction on the depth of the positive proof of $\Gamma \Rightarrow \Delta$ in T_{PT} similarly as before. Because some additional case distinctions are necessary, some additional markers have to be used.

7. Related and current research

The unfolding program founded by Feferman in [17] asks for a given logical system S which operations and predicates ought to be accepted if one accepts the system S. By adding new operation - and predicate symbols to S denoting these operations and predicates often elegant and natural theories are produced. Unfoldings have been presented in the literature for non-finitist and finitist arithmetic (see Feferman [17] and Feferman and Strahm [19, 20]). In Eberhard and Strahm [11], the system T_{PT} plays a crucial role in order to obtain proof-theoretic upper bounds for the full unfolding $\mathcal{U}(\mathsf{FEA})$ of a natural schematic system FEA of feasible arithmetic.

In Cantini [6] interesting additional principles for applicative theories such as choice and uniformity are presented. In his PhD thesis [10], the author addresses extensions of T_{PT} by these principles, and proves that they have polynomial strength too. The difficulty is that the axiom of choice makes the realisation of formulas containing negation necessary. Nevertheless, combining functional realisers as presented in [6] with the realisation dag formalism presented in this paper yields the conservativity result.

In current research, the author addresses the question, which principles of T_{PT} for the truth predicate are necessary to obtain polynomial strength. Obviously, the axioms (C5) and (C6) dealing with the quantifiers are not needed. Interestingly, not even the axiom

$$a \in \mathsf{W} \to (\mathsf{T}(\dot{\mathsf{W}}ab) \leftrightarrow b \leq_{\mathsf{W}} a)$$

does seem to be necessary to prove the totality of all polynomial time computable relations. This is because of the high combinatorial power of the applicative base theory and the flexible truth induction which seem to allow a coding of Turing machine computations.

Results of this kind are of interest because they might suggest new implicit characterisations of well-known complexity classes using different computation principles.

8. Acknowledgements

The author would like to thank the anonymous referees for helpful comments and suggestions that led to significant improvements of this paper. The author would also like to thank Prof. Thomas Strahm and the Swiss National Science Foundation for their support.

References

[1] Aczel, P. Frege structures and the notion of proposition, truth and set. In *The Kleene Symposium* (1980), J. Barwise, H. Keisler, and K. Kunen, Eds., North-Holland, pp. 31– 59.

[2] Beeson, M. J. *Foundations of Constructive Mathematics: Metamathematical Studies*. Springer, Berlin, 1985.

[3] Buss, S. R. *Bounded Arithmetic*. Bibliopolis, Napoli, 1986.

[4] Cantini, A. *Logical Frameworks for Truth and Abstraction*. North-Holland, Amsterdam, 1996.

[5] Cantini, A. Proof-theoretic aspects of self-referential truth. In *Tenth International Congress of Logic, Methodology and Philosophy of Science, Florence, August 1995*, Maria Luisa Dalla Chiara et. al., Ed., vol. 1. Kluwer, September 1997, pp. 7–27.

[6] Cantini, A. Choice and uniformity in weak applicative theories. In *Logic Colloquium '01*, M. Baaz, S. Friedman, and J. Krajíček, Eds., vol. 20 of *Lecture Notes in Logic*. Association for Symbolic Logic, 2005, pp. 108–138.

[7] Clote, P. Computation models and function algebras. In *Handbook of Computability Theory*, E. Griffor, Ed. Elsevier, 1999, pp. 589–681.

[8] Cobham, A. The intrinsic computational difficulty of functions. In *Logic, Methodology and Philosophy of Science II*. North Holland, Amsterdam, 1965, pp. 24–30.

[9] Cook, S. A., and Urquhart, A. Functional interpretations of feasibly constructive arithmetic. *Annals of Pure and Applied Logic 63*, 2 (1993), 103–200.

[10] Eberhard, S. *Weak applicative theories, truth, and computational complexity*. PhD thesis, University of Berne, 2013.

[11] Eberhard, S., and Strahm, T. Unfolding feasible arithmetic and weak truth. In *Axiomatic Theories of Truth* (2012), T. Achourioti, H. Galinon, K. Fujimoto, and J. Martínez-Fernández, Eds., Logic, Epistemology and the Unity of Science, Springer. Being published.

[12] Eberhard, S., and Strahm, T. Weak theories of truth and explicit mathematics. In *Logic, Construction, Computation*, Ulrich Berger, Hannes Diener, and Peter Schuster, Eds. Ontos Verlag, 2012.

[13] Feferman, S. A language and axioms for explicit mathematics. In *Algebra and Logic*, J. Crossley, Ed., vol. 450 of *Lecture Notes in Mathematics*. Springer, Berlin, 1975, pp. 87–139.

[14] Feferman, S. Recursion theory and set theory: a marriage of convenience. In *Generalized recursion theory II, Oslo 1977*, J. E. Fenstad, R. O. Gandy, and G. E. Sacks, Eds., vol. 94 of *Stud. Logic Found. Math.* North Holland, Amsterdam, 1978, pp. 55–98.

[15] Feferman, S. Constructive theories of functions and classes. In *Logic Colloquium '78*, M. Boffa, D. van Dalen, and K. McAloon, Eds. North Holland, Amsterdam, 1979, pp. 159–224.

[16] Feferman, S. Logics for termination and correctness of functional programs. In *Logic from Computer Science*, Y. N. Moschovakis, Ed., vol. 21 of *MSRI Publications*. Springer, Berlin, 1991, pp. 95–127.

[17] Feferman, S. Gödel's program for new axioms: Why, where, how and what? In *Gödel '96*, P. Hájek, Ed., vol. 6 of *Lecture Notes in Logic*. Springer, Berlin, 1996, pp. 3–22.

[18] Feferman, S. Axioms for the determinateness of truth. *Review of Symbolic Logic 1* (2008), 204–217.

[19] Feferman, S., and Strahm, T. The unfolding of non-finitist arithmetic. *Annals of Pure and Applied Logic 104*, 1–3 (2000), 75–96.

[20] Feferman, S., and Strahm, T. Unfolding finitist arithmetic. *Review of Symbolic Logic 3*, 4 (2010), 665–689.

[21] Friedman, H., and Sheard, M. An axiomatic approach to self-referential truth. *Annals of Pure and Applied Logic 33*, 1 (1987), 1–21.

[22] Halbach, V. *Axiomatic Theories of Truth*. Cambridge University Press, 2011.

[23] Kahle, R. Frege structures for partial applicative theories. Tech. Rep. IAM-96-013, Institut für Informatik und angewandte Mathemati, Universität Bern, September 1996.

[24] Kahle, R. *Applikative Theorien und Frege-Strukturen.* PhD thesis, Institut für Informatik und angewandte Mathematik, Universität Bern, 1997.

[25] Kahle, R. *The Applicative Realm.* Habilitation Thesis, Tübingen, 2007. Appeared in Textos de Mathemática 40, Departamento de Mathemática da Universidade de Coimbra, Portugal, 2007.

[26] Probst, D. The provably terminating operations of the subsystem PETJ of explicit mathematics. *Annals of Pure and Applied Logic 162*, 11 (2011), 934–947.

[27] Spescha, D. *Weak systems of explicit mathematics.* PhD thesis, Universität Bern, 2009.

[28] Spescha, D., and Strahm, T. Elementary explicit types and polynomial time operations. *Mathematical Logic Quarterly 55*, 3 (2009), 245–258.

[29] Spescha, D., and Strahm, T. Realizability in weak systems of explicit mathematics. *Mathematical Logic Quarterly 57*, 6 (2011), 551–565.

[30] Strahm, T. *Proof-theoretic Contributions to Explicit Mathematics.* Habilitationsschrift, University of Bern, 2001.

[31] Strahm, T. Theories with self-application and computational complexity. *Information and Computation 185* (2003), 263–297.